DATABASE SYSTEMS

DESIGN IMPLEMENTATION AND MANAGEMENT

INTERNATIONAL EDITION

ROB • CORONEL • CROCKETT

Chapter 6
Advanced Data Modelling

COURSE TECHNOLOGY
CENGAGE Learning™

In this chapter, you will learn:

- About the extended entity relationship (EER) model's main constructs
- How entity clusters are used to represent multiple entities and relationships
- The characteristics of good primary keys and how to select them
- How to use flexible solutions for special data modeling cases
- What issues to check for when developing data models based on EER diagrams

The Extended Entity Relationship Model

- Result of adding more semantic constructs to original entity relationship (ER) model
- Diagram using this model is called an EER diagram (EERD)

COURSE TECHNOLOGY
CENGAGE Learning™

Entity Supertypes and Subtypes

- Entity supertype
  - Generic entity type that is related to one or more entity subtypes
  - Contains common characteristics
- Entity subtypes
  - Contains unique characteristics of each entity subtype

Entity Supertypes and Subtypes (continued)

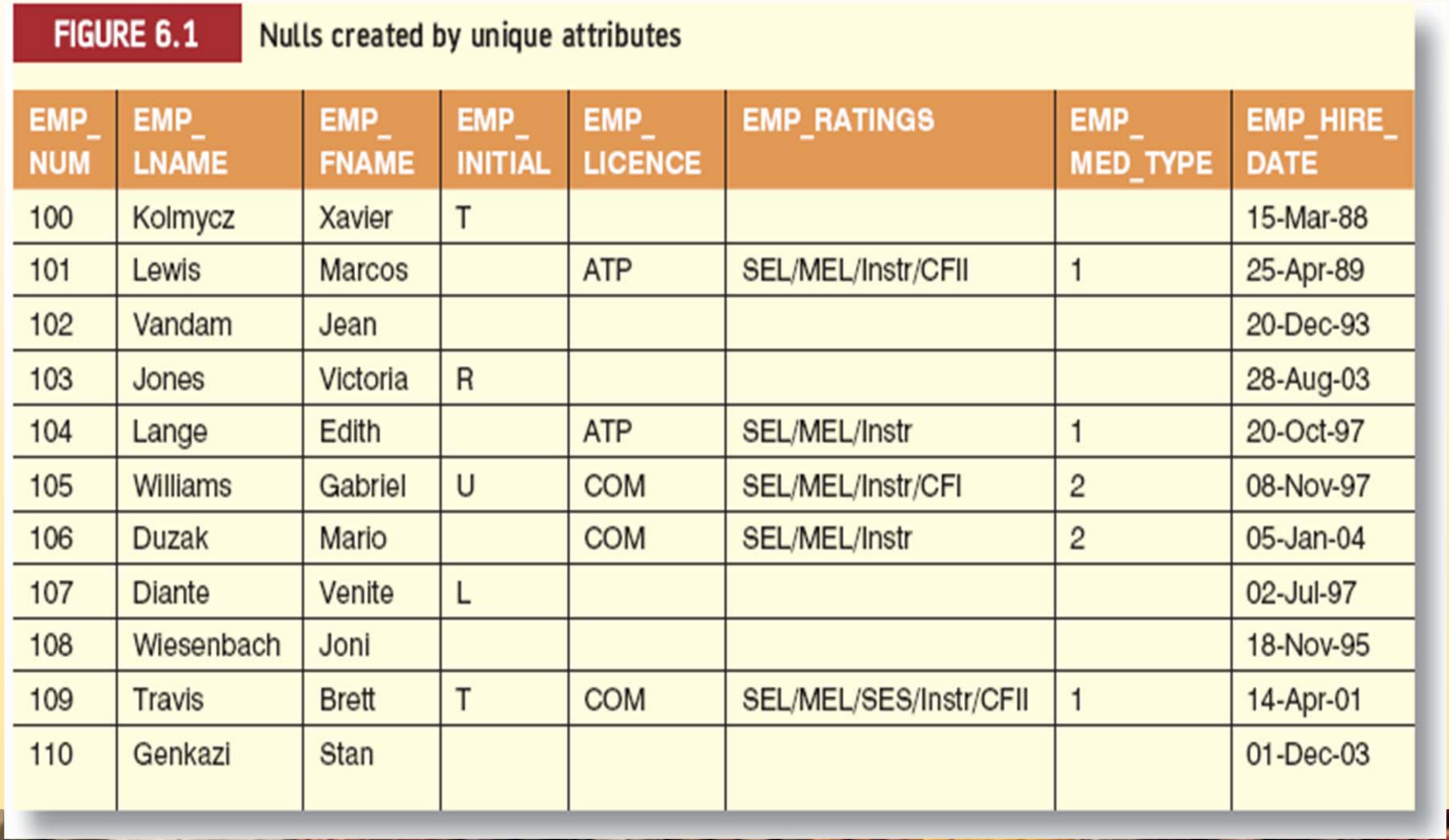


FIGURE 6.1 Nulls created by unique attributes

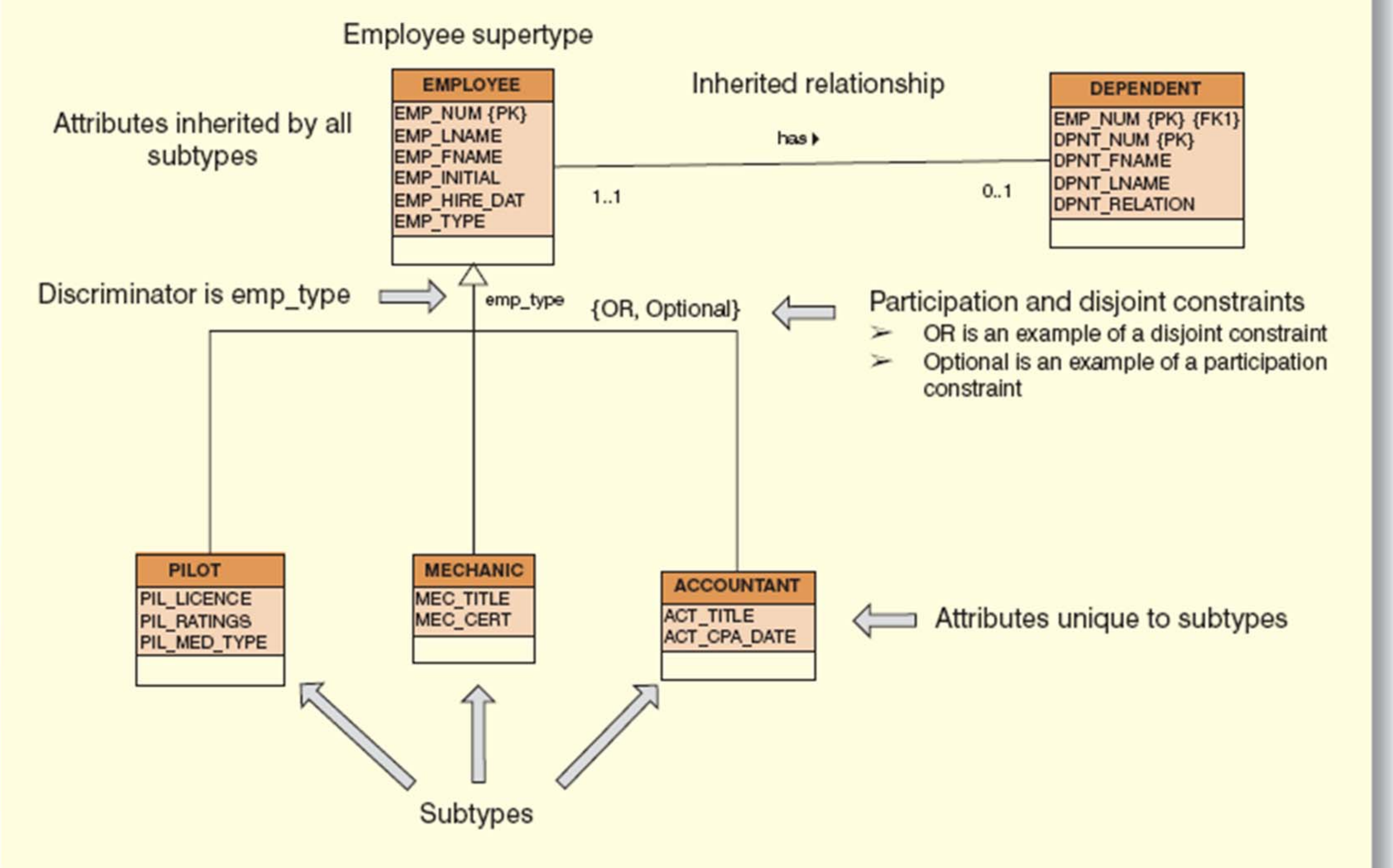
EMP_NUM	EMP_LNAME	EMP_FNAME	EMP_INITIAL	EMP_LICENCE	EMP_RATINGS	EMP_MED_TYPE	EMP_HIRE_DATE
100	Kolmycz	Xavier	T				15-Mar-88
101	Lewis	Marcos		ATP	SEL/MEL/Instr/CFII	1	25-Apr-89
102	Vandam	Jean					20-Dec-93
103	Jones	Victoria	R				28-Aug-03
104	Lange	Edith		ATP	SEL/MEL/Instr	1	20-Oct-97
105	Williams	Gabriel	U	COM	SEL/MEL/Instr/CFI	2	08-Nov-97
106	Duzak	Mario		COM	SEL/MEL/Instr	2	05-Jan-04
107	Diante	Venite	L				02-Jul-97
108	Wiesenbach	Joni					18-Nov-95
109	Travis	Brett	T	COM	SEL/MEL/SES/Instr/CFII	1	14-Apr-01
110	Genkazi	Stan					01-Dec-03

Specialization Hierarchy

- Depicts arrangement of higher-level entity supertypes (parent entities) and lower-level entity subtypes (child entities)
- Relationships sometimes described in terms of “IS-A” relationships
- Subtype can exist only within context of supertype and every subtype can have only one supertype to which it is directly related
- Can have many levels of supertype/subtype relationships

Specialization Hierarchy (continued)

FIGURE 6.2 A specialization hierarchy



Specialization Hierarchy (continued)

- Support attribute inheritance
- Define special supertype attribute known as subtype discriminator
- Define disjoint/overlapping constraints and complete/partial constraints

Inheritance

- Enables entity subtype to inherit attributes and relationships of supertype
- All entity subtypes inherit their primary key attribute from their supertype
- At implementation level, supertype and its subtype(s) depicted in specialization hierarchy maintain a 1:1 relationship

Inheritance (continued)

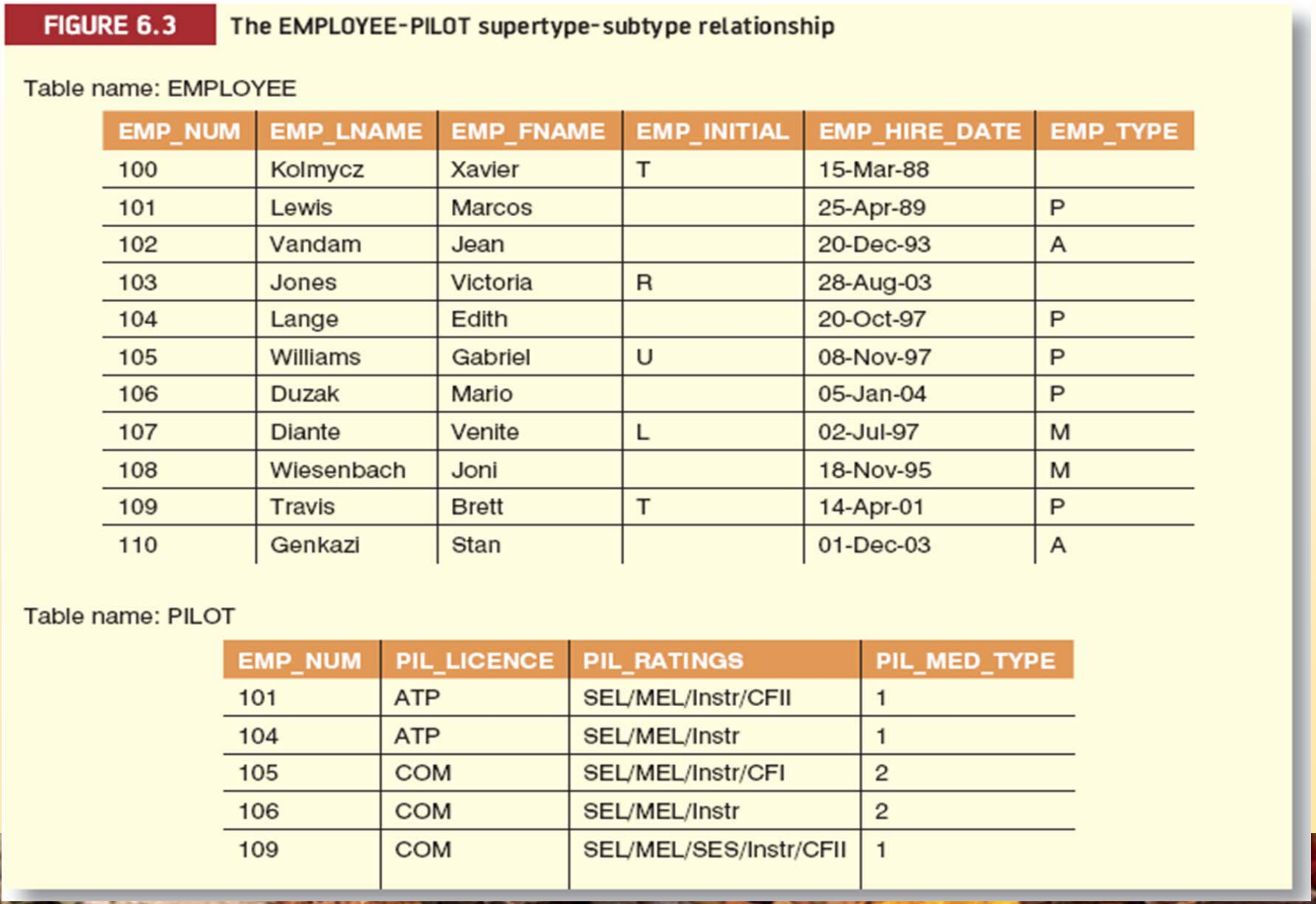


FIGURE 6.3 The EMPLOYEE-PILOT supertype-subtype relationship

Table name: EMPLOYEE

EMP_NUM	EMP_LNAME	EMP_FNAME	EMP_INITIAL	EMP_HIRE_DATE	EMP_TYPE
100	Kolmycz	Xavier	T	15-Mar-88	
101	Lewis	Marcos		25-Apr-89	P
102	Vandam	Jean		20-Dec-93	A
103	Jones	Victoria	R	28-Aug-03	
104	Lange	Edith		20-Oct-97	P
105	Williams	Gabriel	U	08-Nov-97	P
106	Duzak	Mario		05-Jan-04	P
107	Diante	Venite	L	02-Jul-97	M
108	Wiesenbach	Joni		18-Nov-95	M
109	Travis	Brett	T	14-Apr-01	P
110	Genkazi	Stan		01-Dec-03	A

Table name: PILOT

EMP_NUM	PIL_LICENCE	PIL_RATINGS	PIL_MED_TYPE
101	ATP	SEL/MEL/Instr/CFII	1
104	ATP	SEL/MEL/Instr	1
105	COM	SEL/MEL/Instr/CFI	2
106	COM	SEL/MEL/Instr	2
109	COM	SEL/MEL/SES/Instr/CFII	1

COURSE TECHNOLOGY
CENGAGE Learning™

Subtype Discriminator

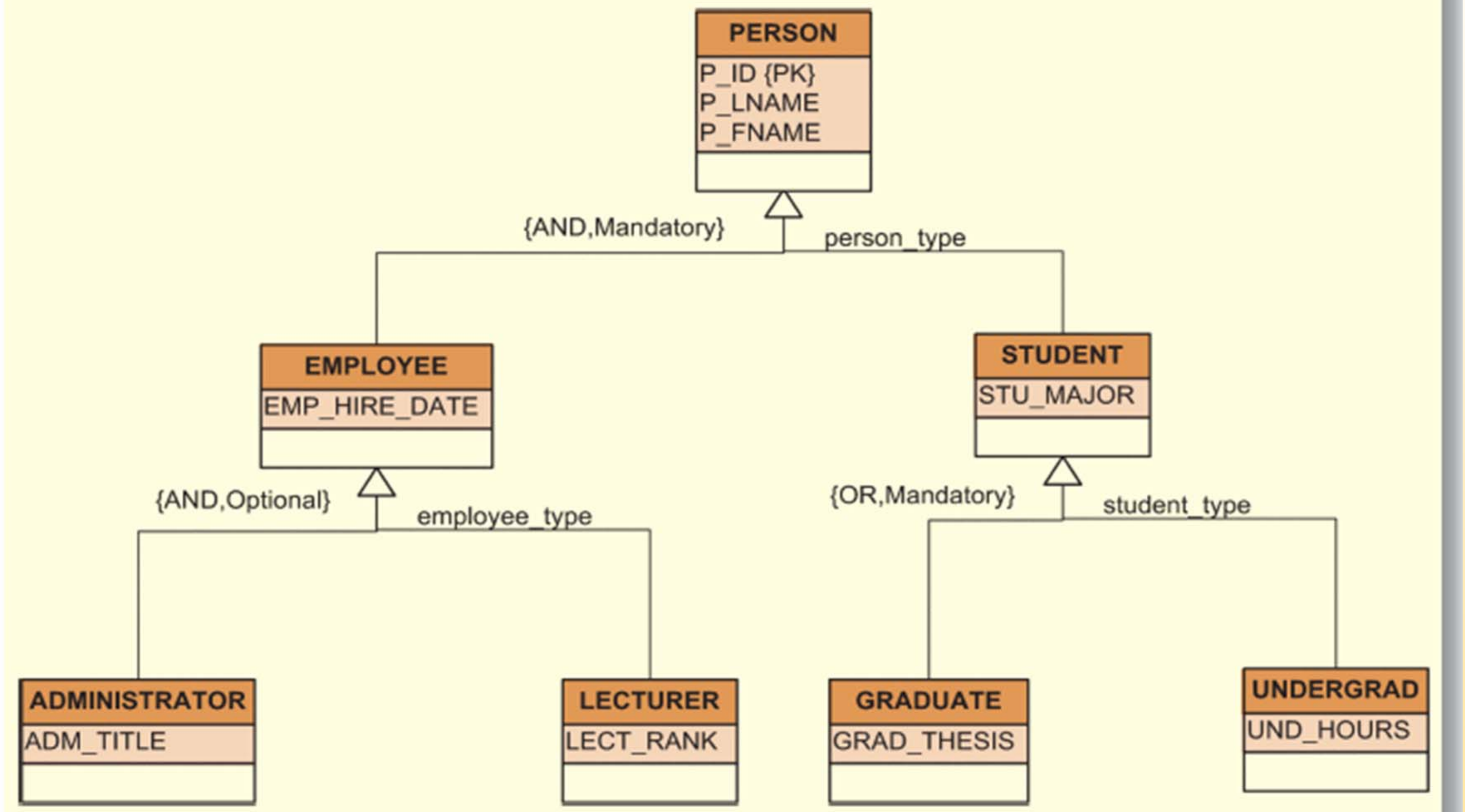
- The attribute in supertype entity that determines to which entity subtype each supertype occurrence is related
- The default comparison condition for subtype discriminator attribute is equality comparison

Disjoint and Overlapping Constraints

- Disjoint subtypes
  - Also known as non-overlapping subtypes
  - Subtypes that contain unique subset of supertype entity set
- Overlapping subtypes
  - Subtypes that contain nonunique subsets of supertype entity set

Disjoint and Overlapping Constraints (cont)

FIGURE 6.4 Specialization hierarchy with overlapping subtypes



Disjoint and Overlapping Constraints (cont)

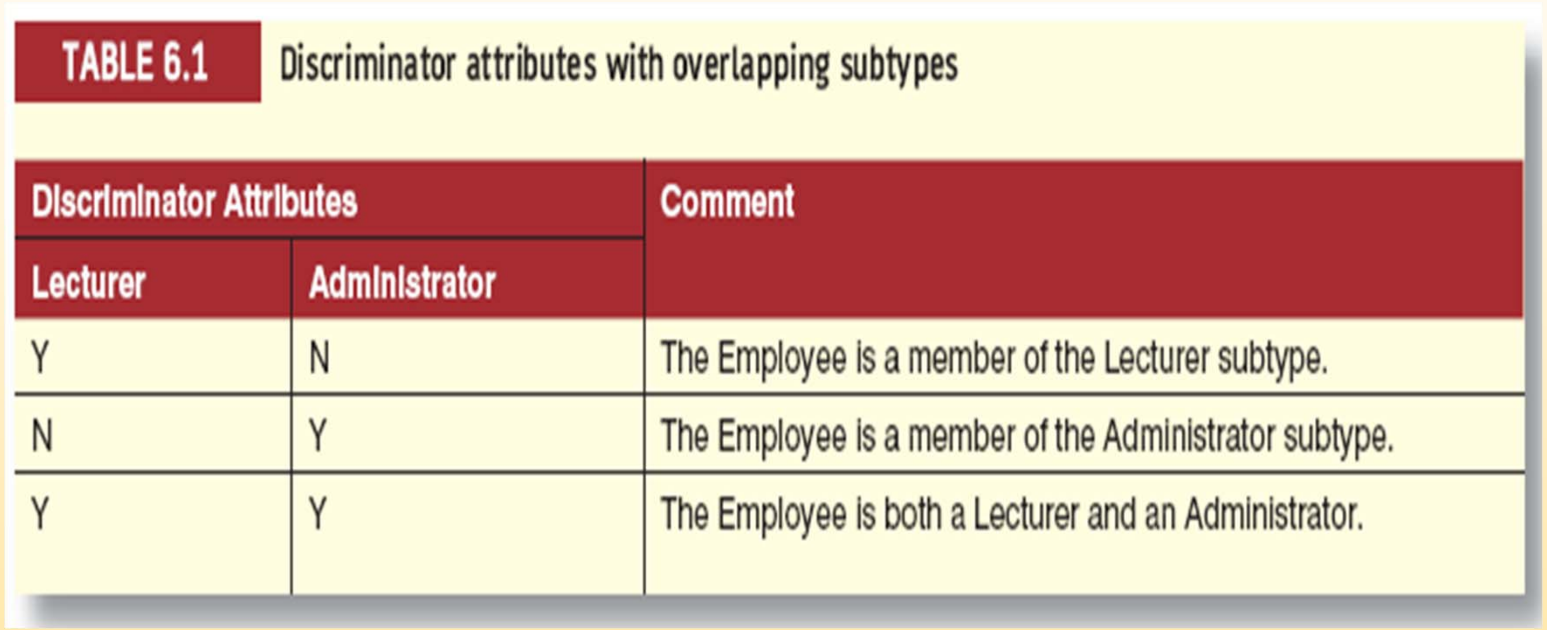


TABLE 6.1 Discriminator attributes with overlapping subtypes

Discriminator Attributes		Comment
Lecturer	Administrator	
Y	N	The Employee is a member of the Lecturer subtype.
N	Y	The Employee is a member of the Administrator subtype.
Y	Y	The Employee is both a Lecturer and an Administrator.

Completeness Constraint

- Specifies whether each entity supertype occurrence must also be member of at least one subtype
- Can be partial or total

Completeness Constraint (continued)

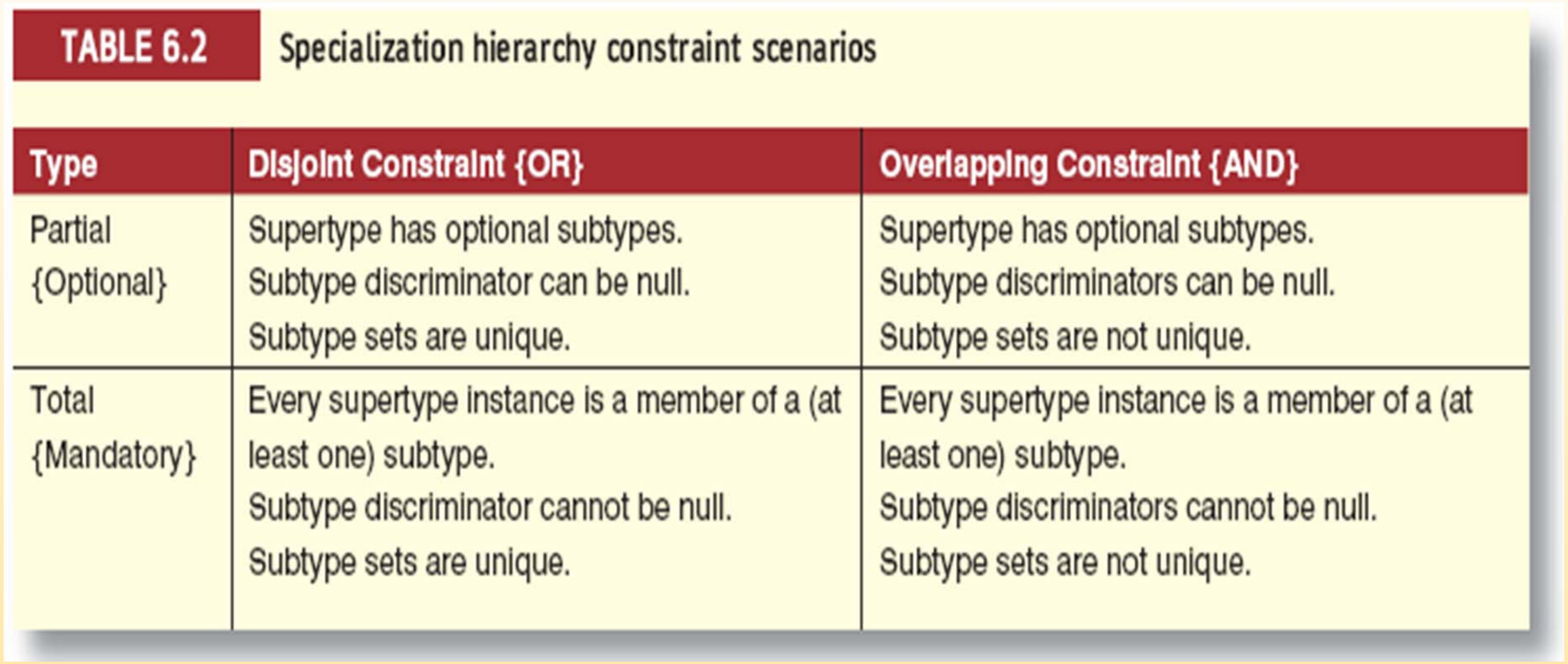


TABLE 6.2 Specialization hierarchy constraint scenarios

Type	Disjoint Constraint {OR}	Overlapping Constraint {AND}
Partial {Optional}	Supertype has optional subtypes. Subtype discriminator can be null. Subtype sets are unique.	Supertype has optional subtypes. Subtype discriminators can be null. Subtype sets are not unique.
Total {Mandatory}	Every supertype instance is a member of a (at least one) subtype. Subtype discriminator cannot be null. Subtype sets are unique.	Every supertype instance is a member of a (at least one) subtype. Subtype discriminators cannot be null. Subtype sets are not unique.

COURSE TECHNOLOGY
CENGAGE Learning™

Specialization and Generalization

- Specialization
  - Top-down process of identifying lower-level, more specific entity subtypes from higher-level entity supertype
  - Based on grouping unique characteristics and relationships of the subtypes

Specialization and Generalization (continued)

- Generalization
  - Bottom-up process of identifying higher-level, more generic entity supertype from lower-level entity subtypes
  - Based on grouping common characteristics and relationships of the subtypes

Composition and Aggregation

- Aggregation
  - a larger entity can be composed of smaller entities.
- Composition
  - special case of aggregation
  - when the parent entity instance is deleted, all child entity instances are automatically deleted.

Composition and Aggregation (continued)

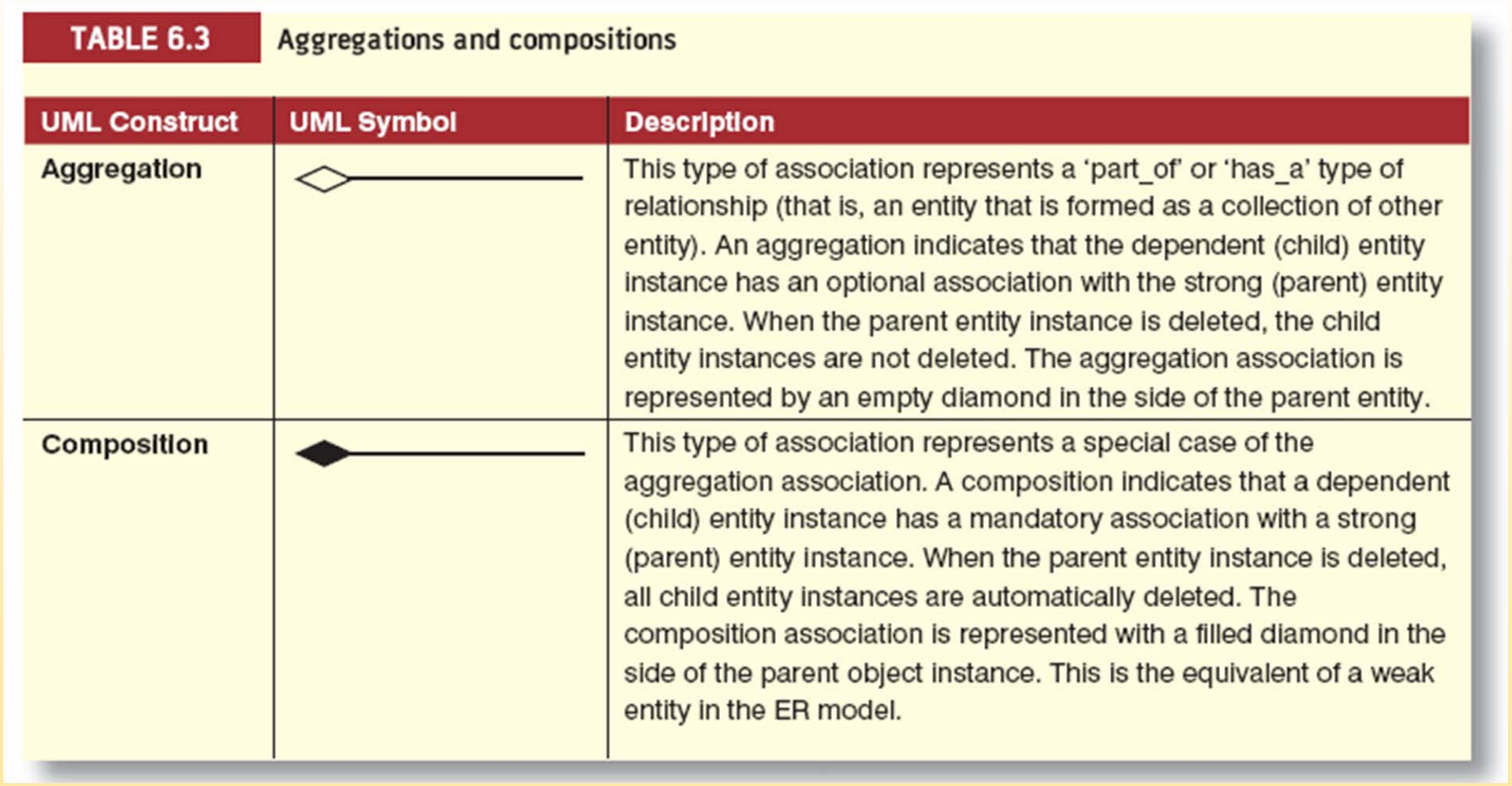


TABLE 6.3 Aggregations and compositions

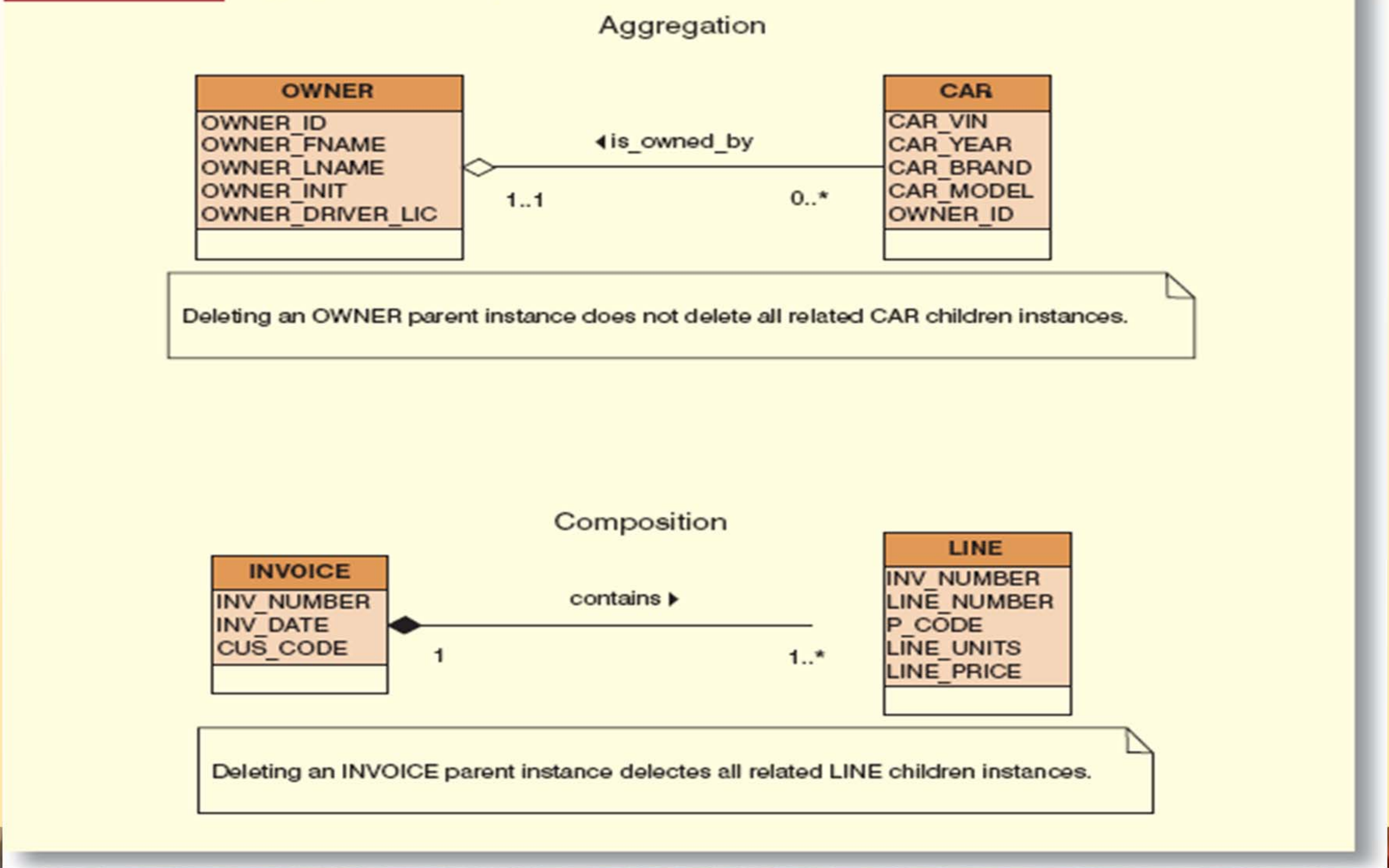
UML Construct	UML Symbol	Description
Aggregation	◇——	This type of association represents a 'part_of' or 'has_a' type of relationship (that is, an entity that is formed as a collection of other entity). An aggregation indicates that the dependent (child) entity instance has an optional association with the strong (parent) entity instance. When the parent entity instance is deleted, the child entity instances are not deleted. The aggregation association is represented by an empty diamond in the side of the parent entity.
Composition	◆——	This type of association represents a special case of the aggregation association. A composition indicates that a dependent (child) entity instance has a mandatory association with a strong (parent) entity instance. When the parent entity instance is deleted, all child entity instances are automatically deleted. The composition association is represented with a filled diamond in the side of the parent object instance. This is the equivalent of a weak entity in the ER model.

Using Aggregation and Composition

- An *aggregation construct* is used when an entity is composed of (or is formed by) a collection of other entities, but the entities are independent of each other.
  - the relationship can be classifi ed as a 'has_a' relationship type.
- A *composition construct* is used when two entities are associated in an aggregation association with a strong identifying relationship.
  - deleting the parent deletes the children instances.

Aggregation and Composition

FIGURE 6.5 Aggregation and composition

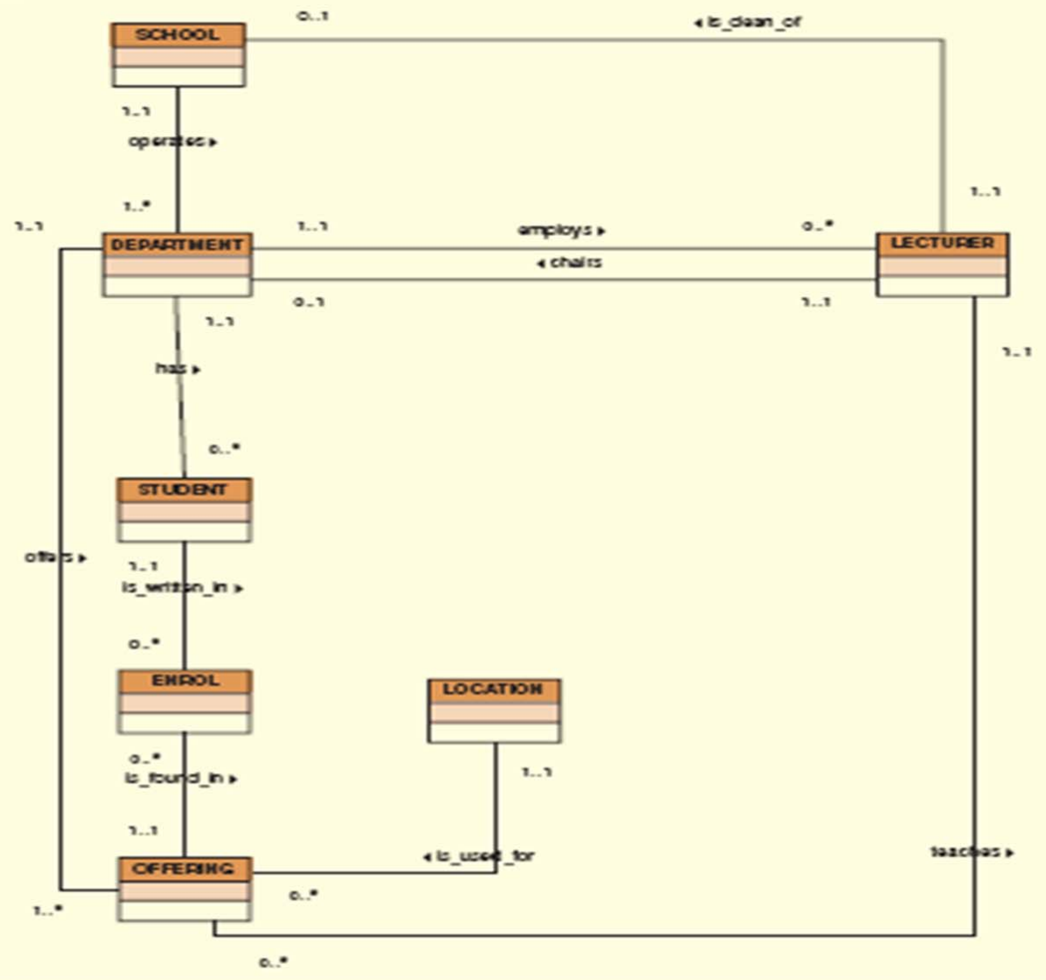


Entity Clustering

- A "virtual" entity type used to represent multiple entities and relationships in ERD
- Considered "virtual" or "abstract" because it is not actually an entity in final ERD
- Temporary entity used to represent multiple entities and relationships
- Eliminate undesirable consequences
  - Avoid display of attributes when entity clusters are used

Entity Clustering (continued)

FIGURE 6.6 Tiny University ERD using entity clusters



Natural Keys and Primary Keys

- Natural key or natural identifier is a real-world, generally accepted identifier used to uniquely identify real-world objects
- Data modeler uses natural identifier as primary key of entity being modeled

Primary Key Guidelines

- Attribute or combination of attributes that uniquely identifies entity instances in an entity set
- Main function is to uniquely identify an entity instance or row within a table
- Guarantee entity integrity, not to “describe” the entity

Primary Key Guidelines (continued)

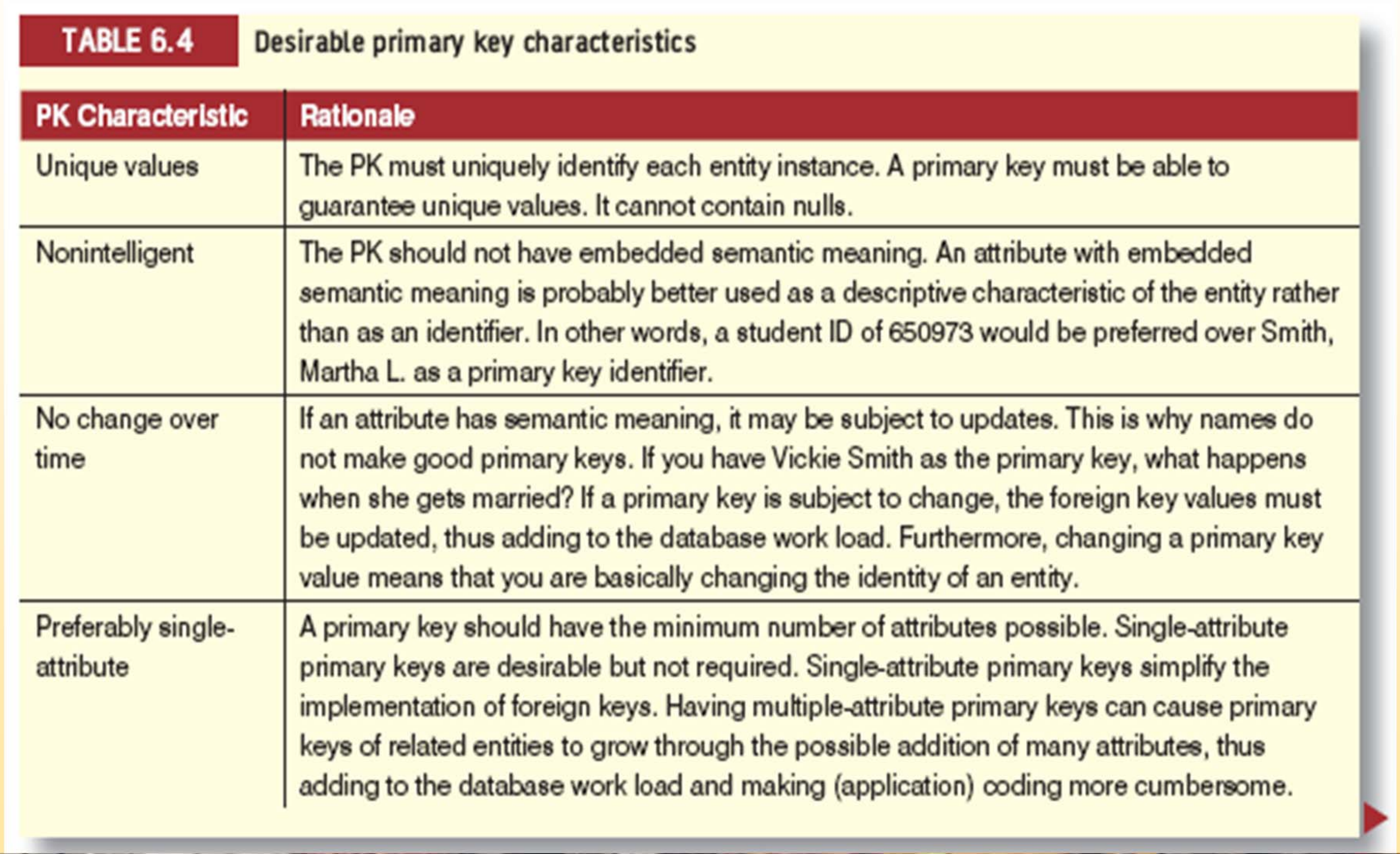
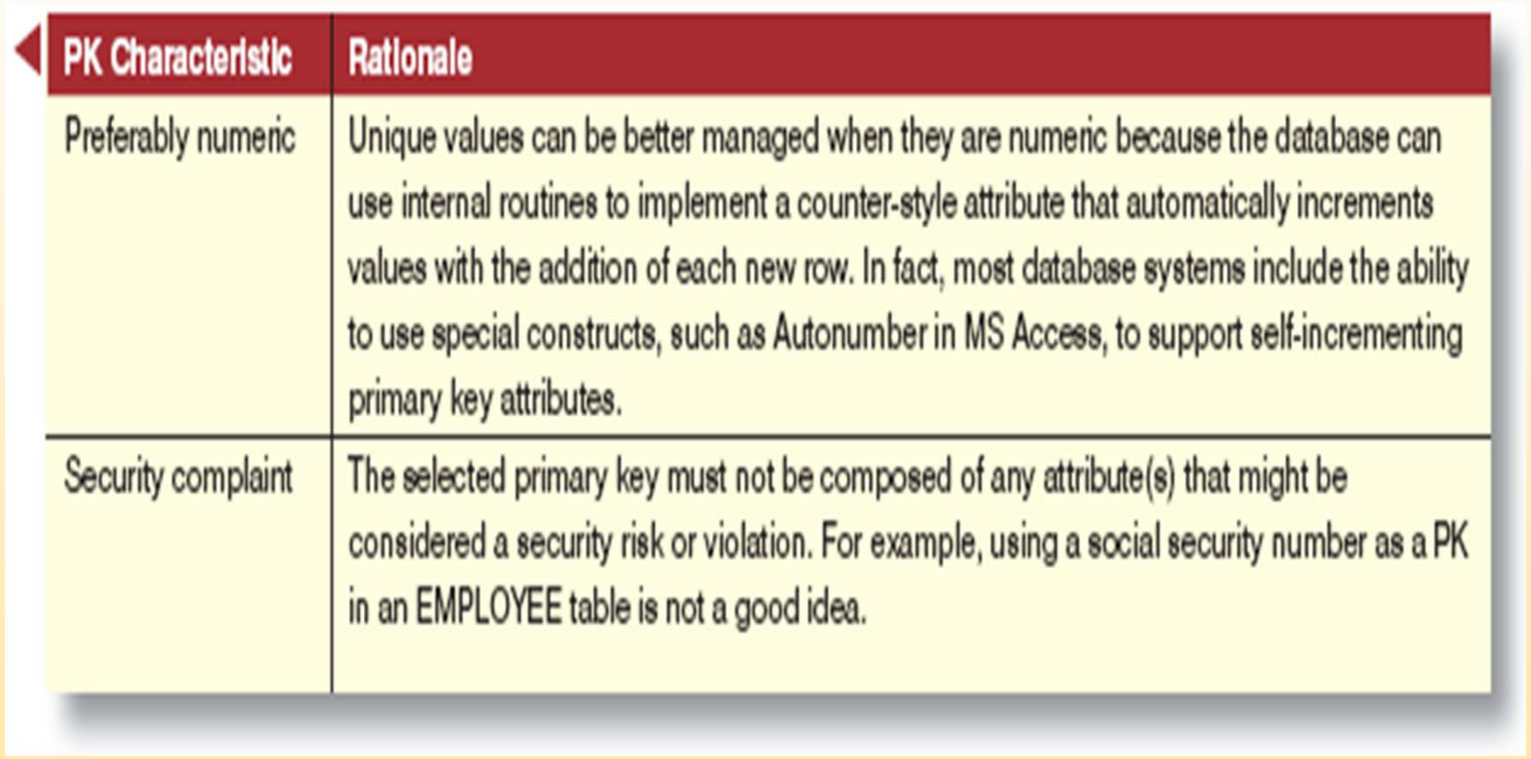


TABLE 6.4 Desirable primary key characteristics

PK Characteristic	Rationale
Unique values	The PK must uniquely identify each entity instance. A primary key must be able to guarantee unique values. It cannot contain nulls.
Nonintelligent	The PK should not have embedded semantic meaning. An attribute with embedded semantic meaning is probably better used as a descriptive characteristic of the entity rather than as an identifier. In other words, a student ID of 650973 would be preferred over Smith, Martha L. as a primary key identifier.
No change over time	If an attribute has semantic meaning, it may be subject to updates. This is why names do not make good primary keys. If you have Vickie Smith as the primary key, what happens when she gets married? If a primary key is subject to change, the foreign key values must be updated, thus adding to the database work load. Furthermore, changing a primary key value means that you are basically changing the identity of an entity.
Preferably single-attribute	A primary key should have the minimum number of attributes possible. Single-attribute primary keys are desirable but not required. Single-attribute primary keys simplify the implementation of foreign keys. Having multiple-attribute primary keys can cause primary keys of related entities to grow through the possible addition of many attributes, thus adding to the database work load and making (application) coding more cumbersome.

Primary Key Guidelines (continued)



PK Characteristic	Rationale
Preferably numeric	Unique values can be better managed when they are numeric because the database can use internal routines to implement a counter-style attribute that automatically increments values with the addition of each new row. In fact, most database systems include the ability to use special constructs, such as Autonumber in MS Access, to support self-incrementing primary key attributes.
Security complaint	The selected primary key must not be composed of any attribute(s) that might be considered a security risk or violation. For example, using a social security number as a PK in an EMPLOYEE table is not a good idea.

COURSE TECHNOLOGY
CENGAGE Learning™

When to Use Composite Primary Keys

- Useful as identifiers of composite entities, where each primary key combination is allowed only once in *:* relationship
  - Automatically provides benefit of ensuring that there cannot be duplicate values

When to Use Composite Primary Keys (cont)

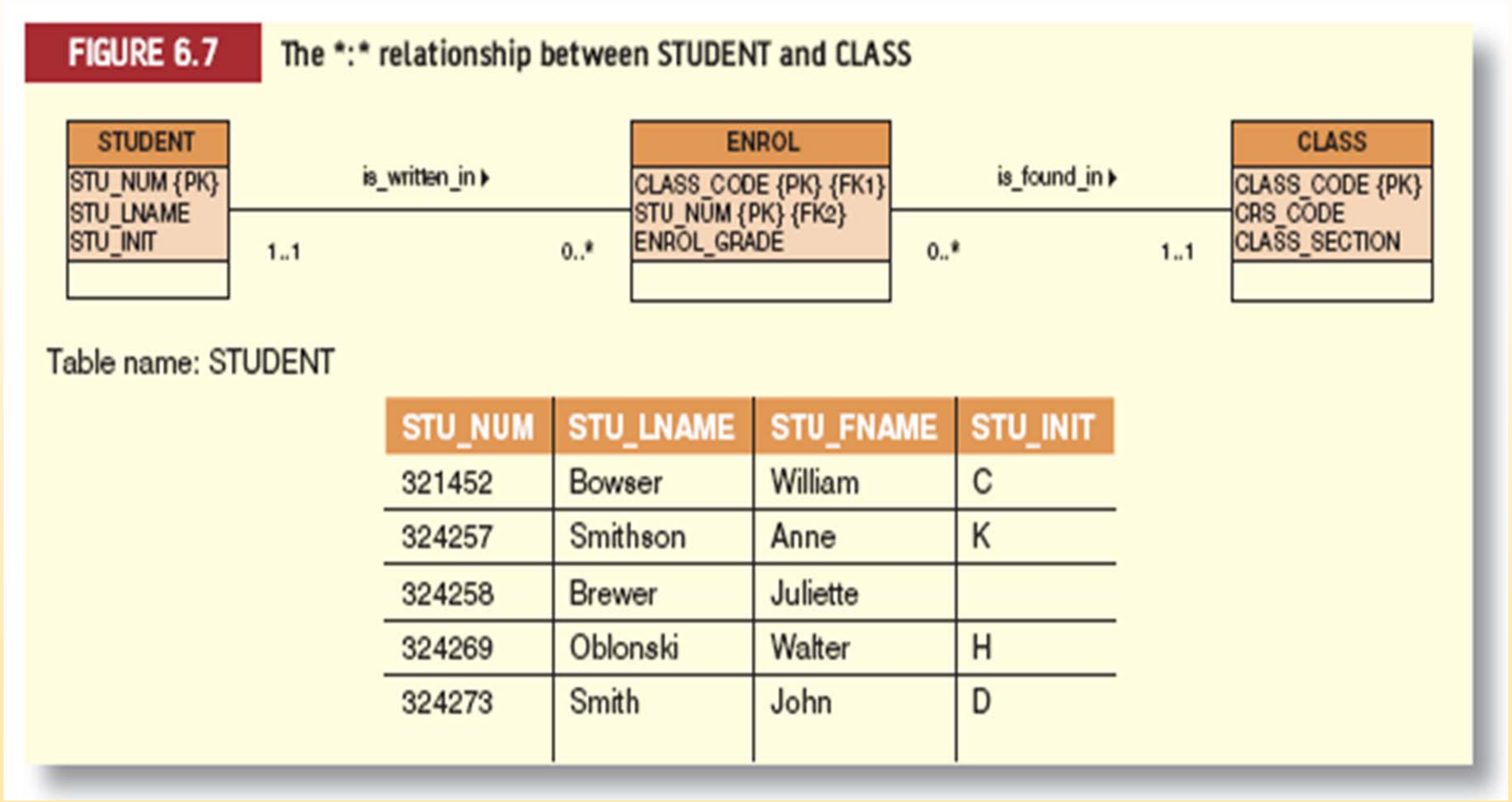
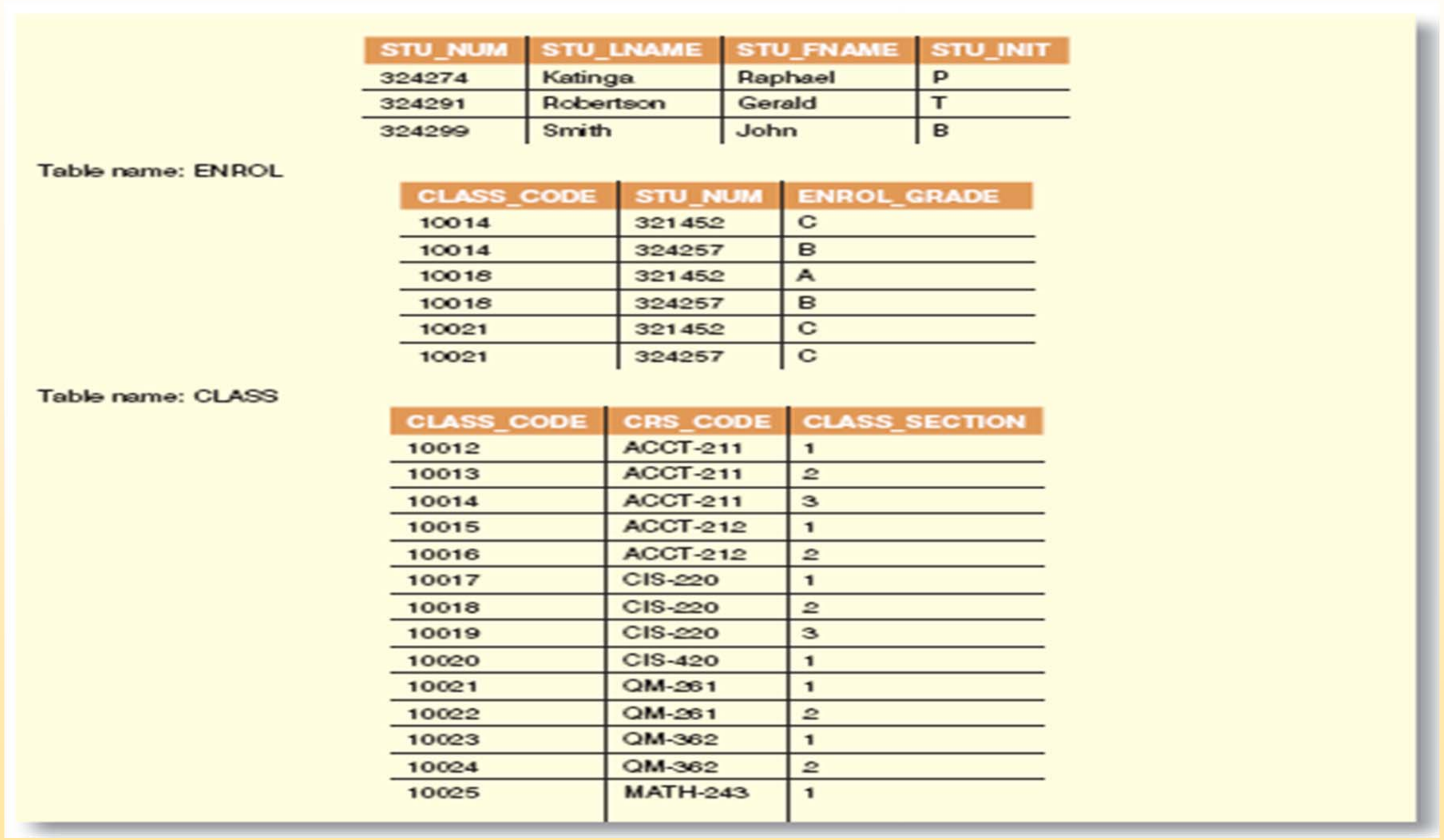


FIGURE 6.7 The *:* relationship between STUDENT and CLASS

Table name: STUDENT

STU_NUM	STU_LNAME	STU_FNAME	STU_INIT
321452	Bowser	William	C
324257	Smithson	Anne	K
324258	Brewer	Juliette	
324269	Oblonski	Walter	H
324273	Smith	John	D

When to Use Composite Primary Keys (cont)



STU_NUM	STU_LNAME	STU_FNAME	STU_INIT
324274	Katinga	Raphael	P
324291	Robertson	Gerald	T
324299	Smith	John	B

Table name: ENROL

CLASS_CODE	STU_NUM	ENROL_GRADE
10014	321452	C
10014	324257	B
10018	321452	A
10018	324257	B
10021	321452	C
10021	324257	C

Table name: CLASS

CLASS_CODE	CRS_CODE	CLASS_SECTION
10012	ACCT-211	1
10013	ACCT-211	2
10014	ACCT-211	3
10015	ACCT-212	1
10016	ACCT-212	2
10017	CIS-220	1
10018	CIS-220	2
10019	CIS-220	3
10020	CIS-420	1
10021	QM-261	1
10022	QM-261	2
10023	QM-362	1
10024	QM-362	2
10025	MATH-243	1

When to Use Composite Primary Keys (continued)

- Useful as identifiers of weak entities, where weak entity has strong identifying relationship with parent entity
  - Normally used to represent:
    - A real-world object that is existent dependent on another real-world object
    - A real-world object that is represented in data model as two separate entities in strong identifying relationship

COURSE TECHNOLOGY
CENGAGE Learning™

When To Use Surrogate Primary Keys

- Especially helpful when there is:
  - No natural key
  - Selected candidate key has embedded semantic contents
  - Selected candidate key is too long or cumbersome
- If you use surrogate key, ensure that candidate key of entity in question performs properly through use of “unique index” and “not null” constraints

When To Use Surrogate Primary Keys (continued)

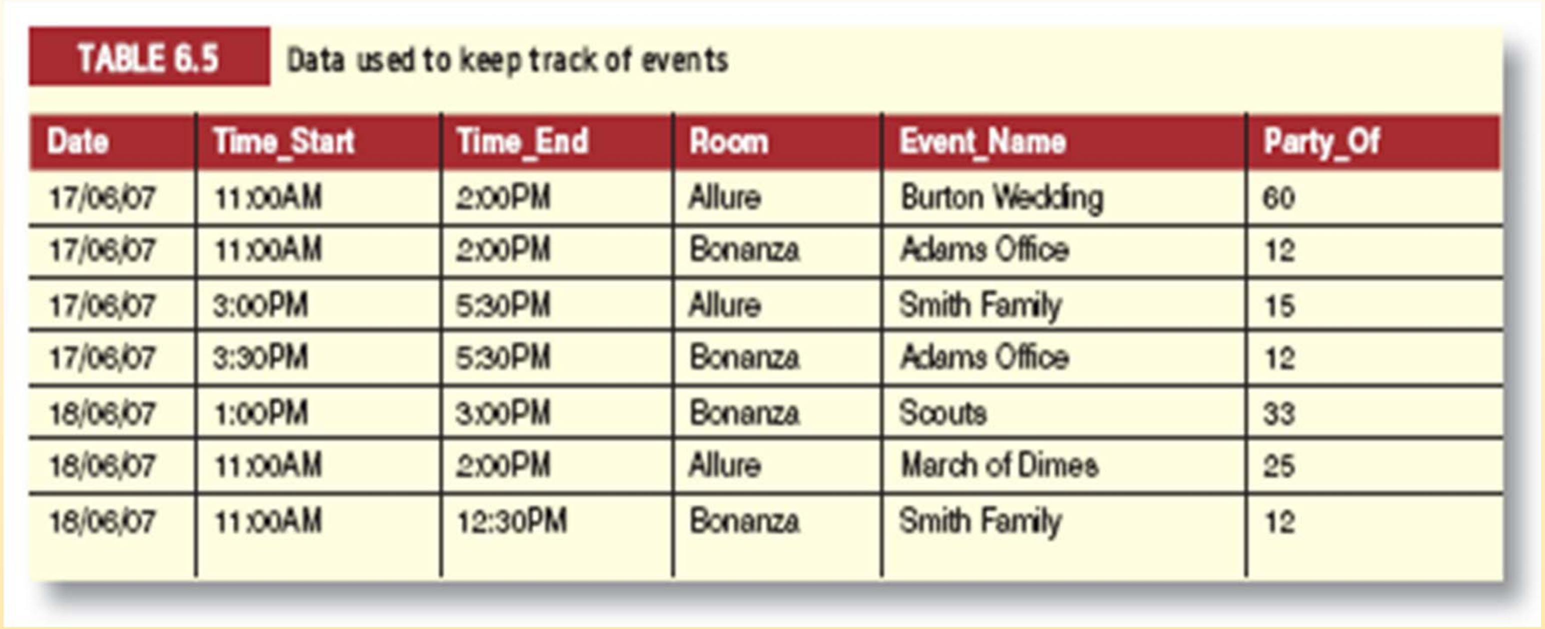


TABLE 6.5 Data used to keep track of events

Date	Time_Start	Time_End	Room	Event_Name	Party_Of
17/06/07	11:00AM	2:00PM	Allure	Burton Wedding	60
17/06/07	11:00AM	2:00PM	Bonanza	Adams Office	12
17/06/07	3:00PM	5:30PM	Allure	Smith Family	15
17/06/07	3:30PM	5:30PM	Bonanza	Adams Office	12
18/06/07	1:00PM	3:00PM	Bonanza	Scouts	33
18/06/07	11:00AM	2:00PM	Allure	March of Dimes	25
18/06/07	11:00AM	12:30PM	Bonanza	Smith Family	12

COURSE TECHNOLOGY
CENGAGE Learning™

Design Case #1:
Implementing 1:1 Relationships

- Foreign keys work with primary keys to properly implement relationships in relational model
- Put primary key of the "one" side (parent entity) on the "many" side (dependent entity) as foreign key
- A 1:1 relationship is used to ensure that two entity sets are not placed in same table

COURSE TECHNOLOGY
CENGAGE Learning™

Design Case #1: Implementing 1:1 Relationships (continued)

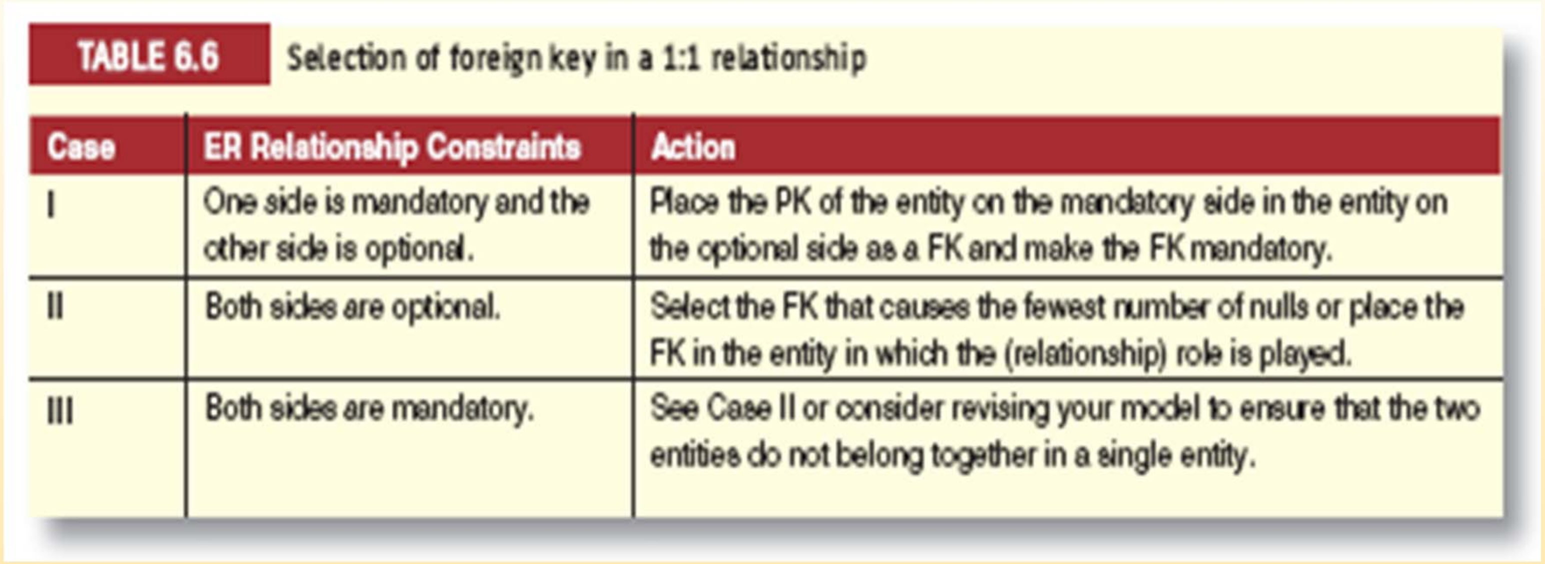
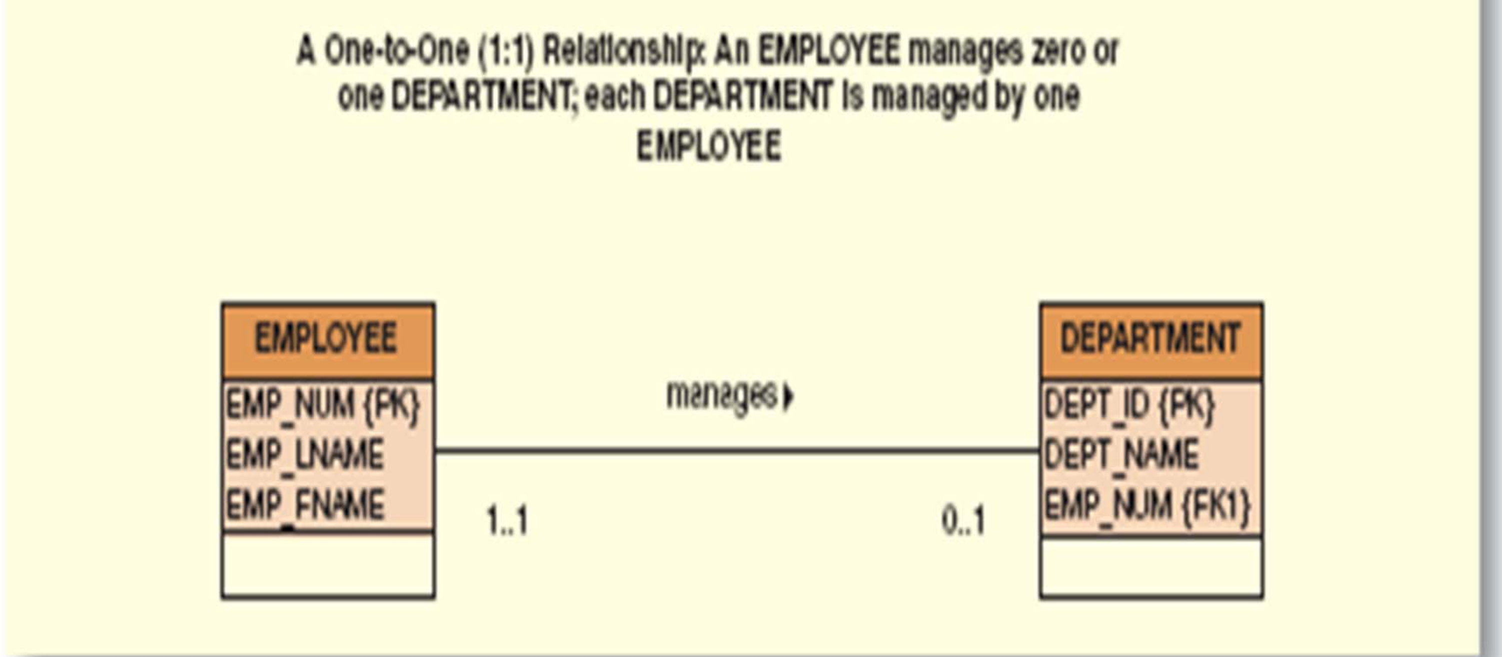


TABLE 6.6 Selection of foreign key in a 1:1 relationship

Case	ER Relationship Constraints	Action
I	One side is mandatory and the other side is optional.	Place the PK of the entity on the mandatory side in the entity on the optional side as a FK and make the FK mandatory.
II	Both sides are optional.	Select the FK that causes the fewest number of nulls or place the FK in the entity in which the (relationship) role is played.
III	Both sides are mandatory.	See Case II or consider revising your model to ensure that the two entities do not belong together in a single entity.

Design Case #1: Implementing 1:1 Relationships (continued)

FIGURE 6.8 A 1:1 relationship between DEPARTMENT and EMPLOYEE

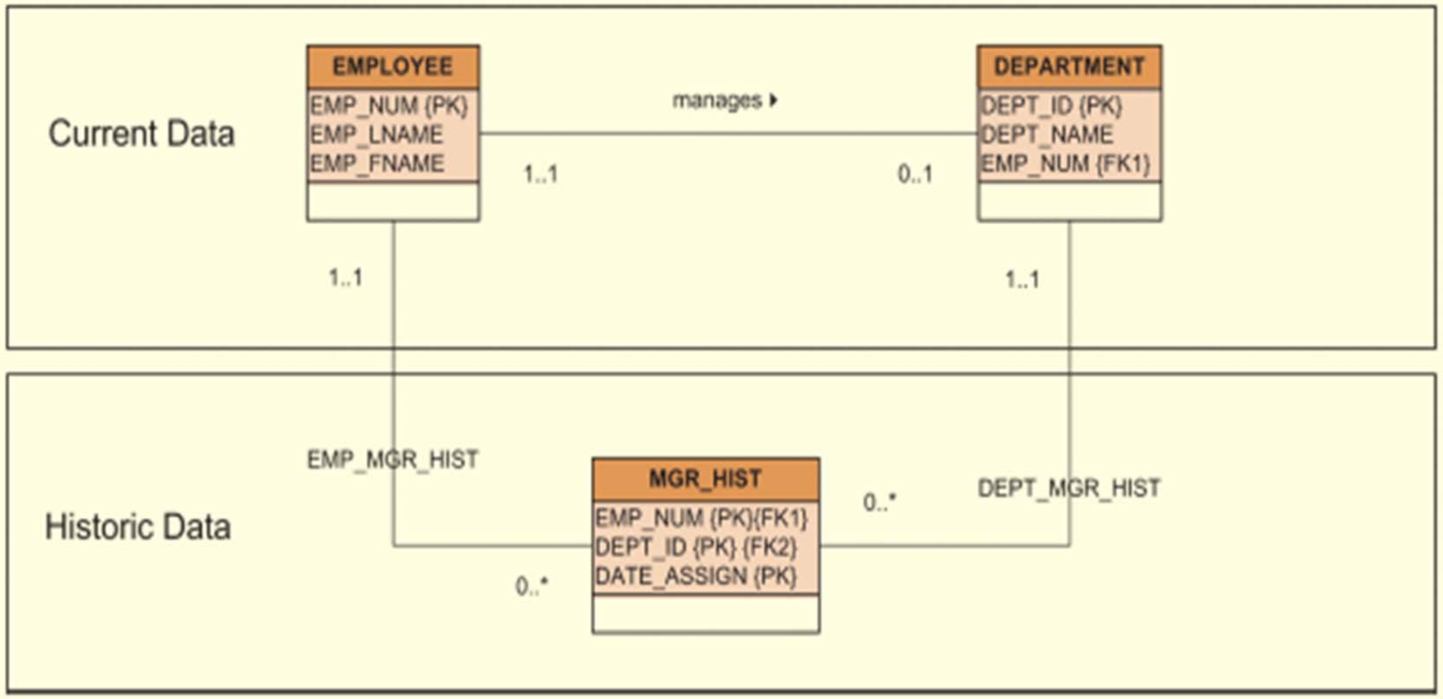

A One-to-One (1:1) Relationship: An EMPLOYEE manages zero or one DEPARTMENT; each DEPARTMENT is managed by one EMPLOYEE
EMPLOYEE
EMP_NUM {PK}
EMP_LNAME
EMP_FNAME
manages ▸
1..1
0..1
DEPARTMENT
DEPT_ID {PK}
DEPT_NAME
EMP_NUM {FK1}

Design Case #2: Maintaining History of Time-Variant Data

- Time-variant data refers to data whose values change over time and for which you must keep a history of data changes

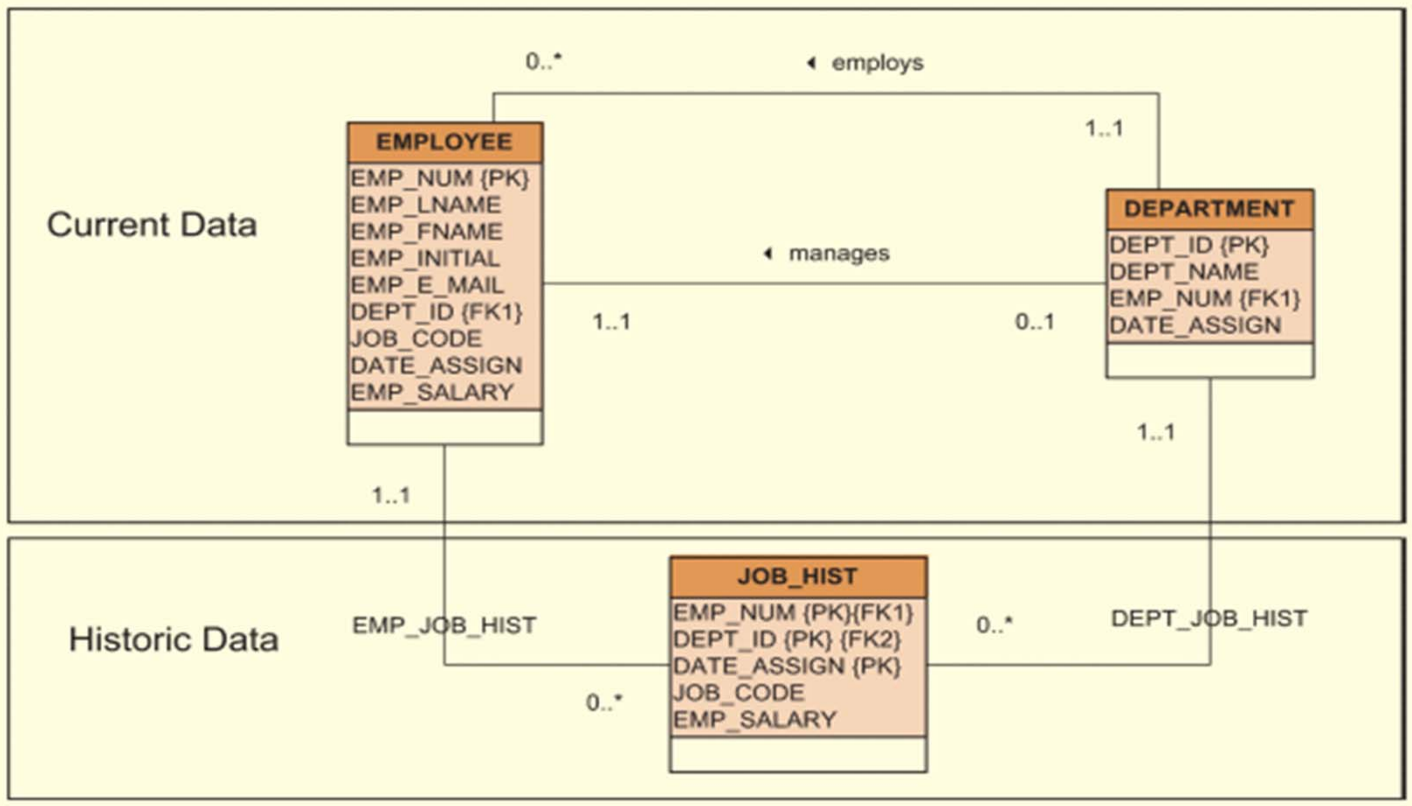
Design Case #2: Maintaining History of Time-Variant Data (continued)

FIGURE 6.9 Maintaining manager history



Design Case #2: Maintaining History of Time-Variant Data (continued)

FIGURE 6.10 Maintaining job history

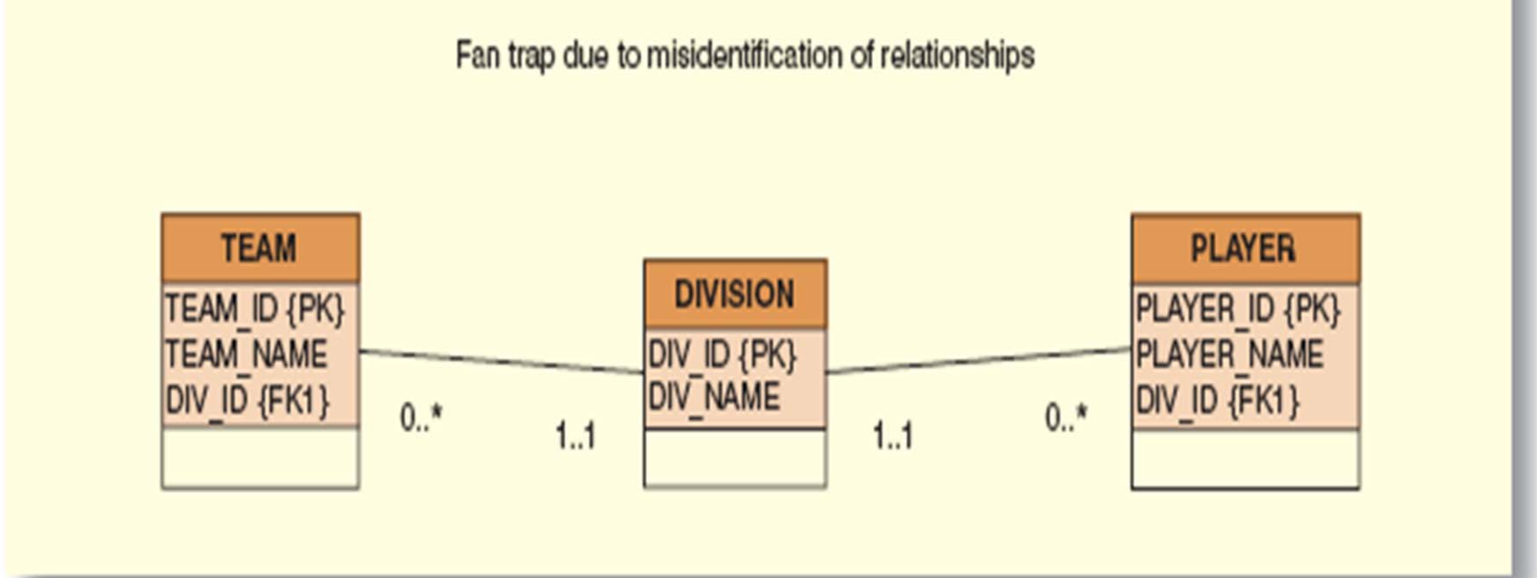


Design Case #3: Fan Traps

- Design trap occurs when relationship is improperly or incompletely identified
- Most common design trap is known as fan trap
- Fan trap occurs when having one entity in two 1:* relationships to other entities
  - Thus producing an association among other entities that is not expressed in model

Design Case #3: Fan Traps (continued)

FIGURE 6.11 Incorrect ERD with fan trap problem



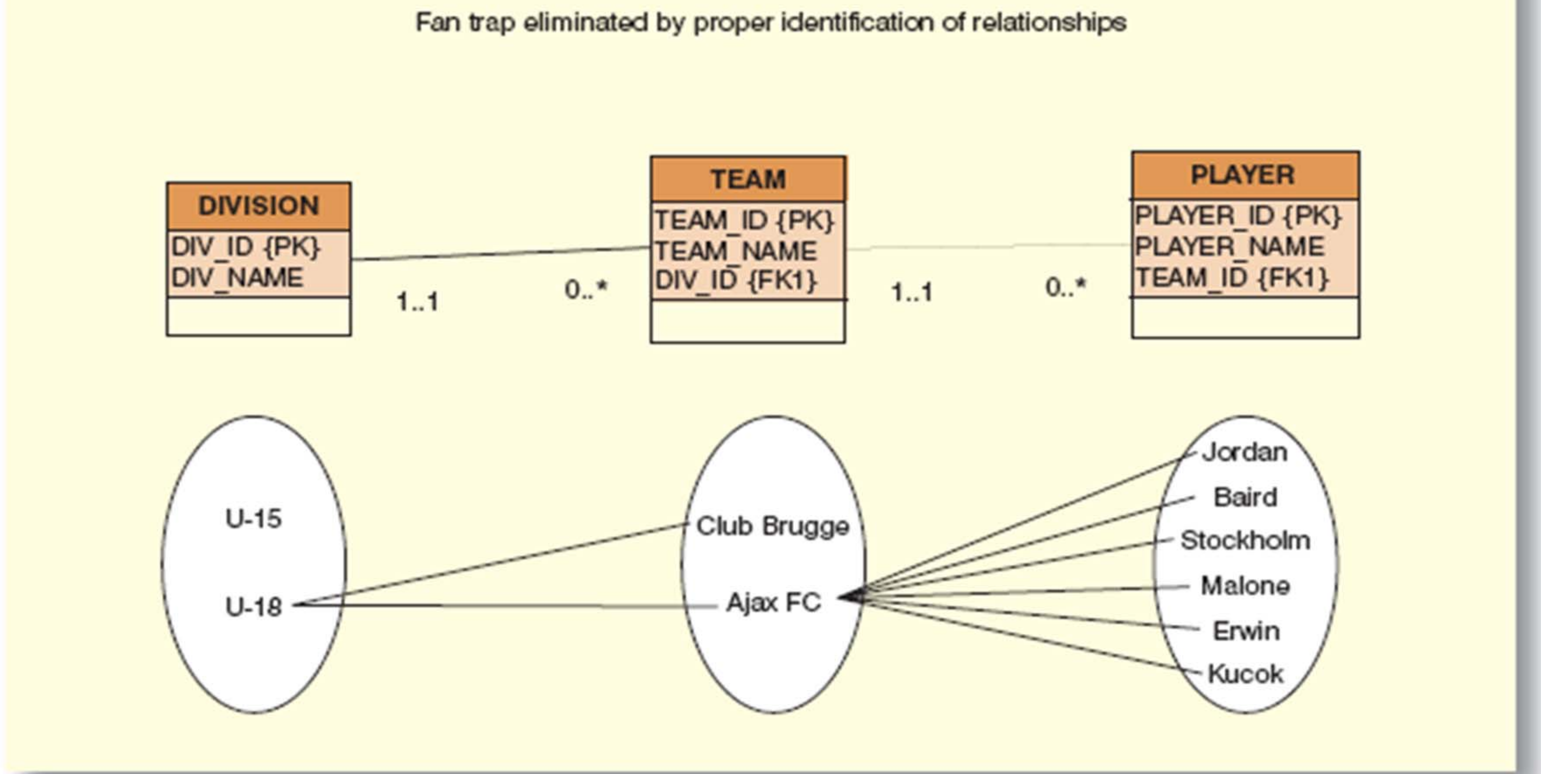
Design Case #4:
Redundant Relationships

- Redundancy is seldom a good thing in database environment
- Occur when there are multiple relationship paths between related entities
- Main concern is that redundant relationships remain consistent across model

COURSE TECHNOLOGY
CENGAGE Learning™

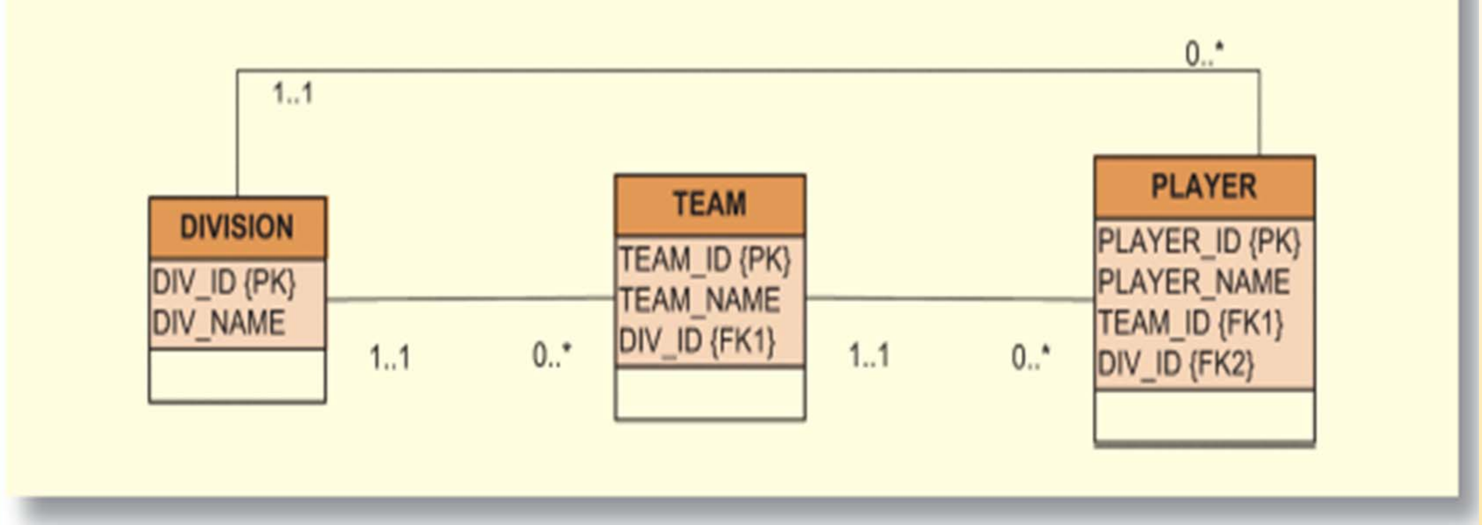
Design Case #4: Redundant Relationships (continued)

FIGURE 6.12 Corrected ERD after removal of the fan trap



Design Case #4: Redundant Relationships (continued)

FIGURE 6.13 A redundant relationship



Data Modeling Checklist

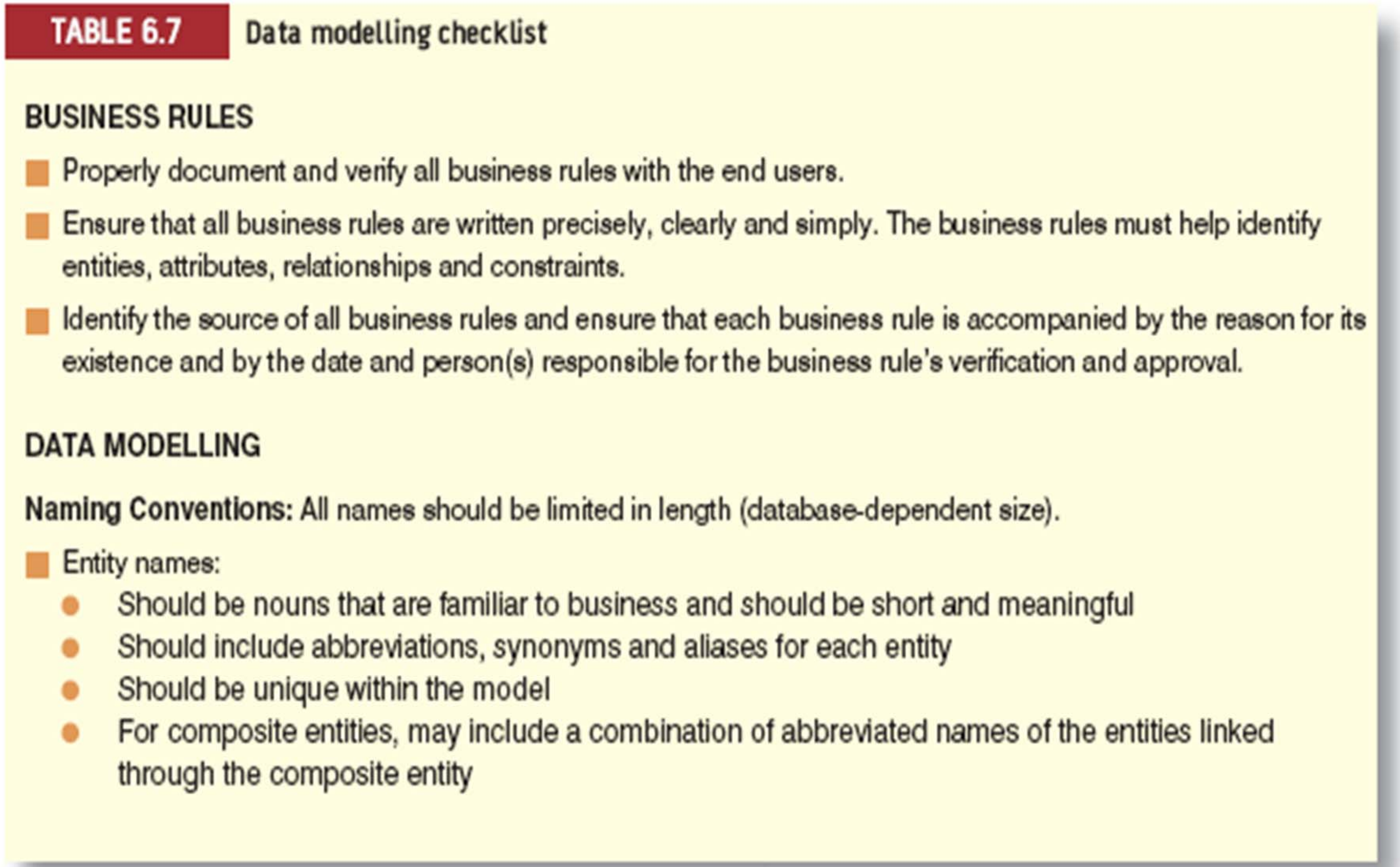


TABLE 6.7 Data modelling checklist

BUSINESS RULES

- Properly document and verify all business rules with the end users.
- Ensure that all business rules are written precisely, clearly and simply. The business rules must help identify entities, attributes, relationships and constraints.
- Identify the source of all business rules and ensure that each business rule is accompanied by the reason for its existence and by the date and person(s) responsible for the business rule's verification and approval.

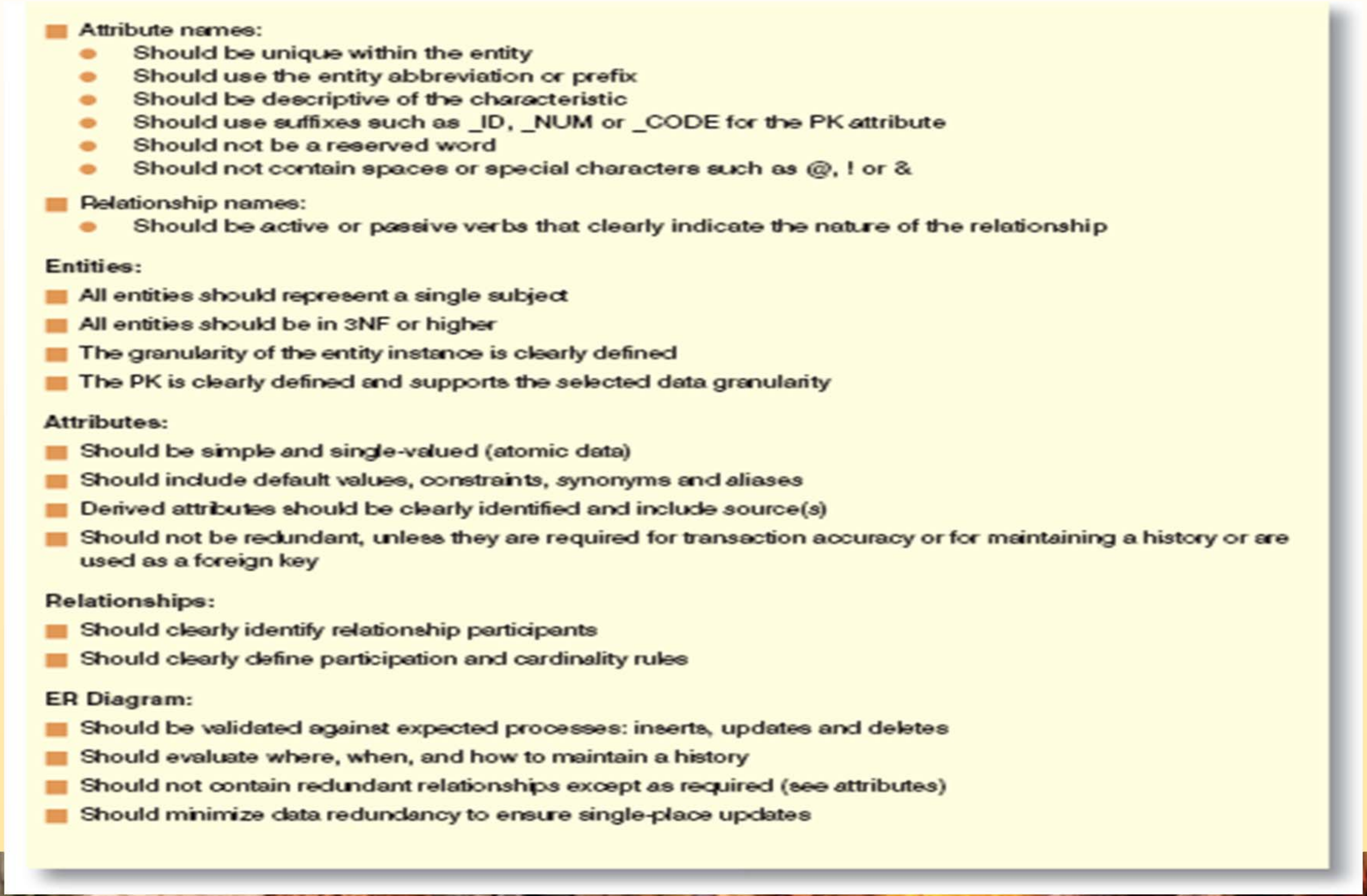
DATA MODELLING

Naming Conventions: All names should be limited in length (database-dependent size).

- Entity names:
  - Should be nouns that are familiar to business and should be short and meaningful
  - Should include abbreviations, synonyms and aliases for each entity
  - Should be unique within the model
  - For composite entities, may include a combination of abbreviated names of the entities linked through the composite entity

COURSE TECHNOLOGY
CENGAGE Learning™

Data Modeling Checklist (continued)



- Attribute names:
  - Should be unique within the entity
  - Should use the entity abbreviation or prefix
  - Should be descriptive of the characteristic
  - Should use suffixes such as _ID, _NUM or _CODE for the PK attribute
  - Should not be a reserved word
  - Should not contain spaces or special characters such as @, ! or &
- Relationship names:
  - Should be active or passive verbs that clearly indicate the nature of the relationship

Entities:

- All entities should represent a single subject
- All entities should be in 3NF or higher
- The granularity of the entity instance is clearly defined
- The PK is clearly defined and supports the selected data granularity

Attributes:

- Should be simple and single-valued (atomic data)
- Should include default values, constraints, synonyms and aliases
- Derived attributes should be clearly identified and include source(s)
- Should not be redundant, unless they are required for transaction accuracy or for maintaining a history or are used as a foreign key

Relationships:

- Should clearly identify relationship participants
- Should clearly define participation and cardinality rules

ER Diagram:

- Should be validated against expected processes: inserts, updates and deletes
- Should evaluate where, when, and how to maintain a history
- Should not contain redundant relationships except as required (see attributes)
- Should minimize data redundancy to ensure single-place updates

Summary

- Extended entity relationship (EER) model adds semantics to ER model via entity supertypes, subtypes, and clusters
- Specialization hierarchy depicts arrangement and relationships between entity supertypes and entity subtypes
- Inheritance allows an entity subtype to inherit attributes and relationships of supertype

Summary (continued)

- Entity cluster is “virtual” entity type used to represent multiple entities and relationships in ERD
- Natural keys are identifiers that exist in real world
- Composite keys are useful to represent *:* relationships and weak (strong-identifying) entities

Summary (continued)

- Surrogate primary keys are useful when there is no natural key that makes a suitable primary key
- In a 1:1 relationship, place the PK of mandatory entity as foreign key in optional entity
- Time-variant data refers to data whose values change over time and whose requirements mandate that you keep a history of data changes

Summary (continued)

- Fan trap occurs when you have one entity in two 1:* relationships to other entities and there is an association among the other entities that is not expressed in model
- Data modeling checklist provides way for designer to check that the ERD meets set of minimum requirements